MACOSA – Assembler für C 8205/C 8205 Z

Jochen Focke
Martin-Luther-Universität Halle

Die Eingabeteile der Interpretiersysteme setzen numerische – absolute oder relative – Adreßteile der Befehle voraus. Bei der Programmierung muß die Stellung der Operanden und Befehle zueinander durch Durchnumerieren der Befehlsfolge festgelegt werden, ein nachträgliches Streichen oder Einfügen ist nicht möglich. Die Erfüllung dieser Forderungen ist mit großem Zeitaufwand und einer Reihe von Schwierigkeiten und Fehlermöglichkeiten verbunden. Außerdem ist die Einführung von Nummern für die Operanden von der Problemseite her völlig willkürlich. Es ist daher zweckmäßig, die numerischen Adreßteile der Befehle durch sinnfällige Bezeichnungen zu ersetzen, die andererseits als Namen vor den anzusprechenden Operanden und Befehlen erscheinen. Die Quellensprache ‚macosa' (Maschinencode mit symbolischer Adressierung) läßt dies zu. Um die symbolischen Adreßteile als solche zu charakterisieren, steht in diesen Fällen nach dem ‚Ersten Kennzeichen' K1 = 0 das ‚Zweite Kennzeichen' K2 = s. Vom Assembler werden die symbolischen Adreßteile durch numerische ersetzt. Die erzeugten Objektprogramme entsprechen in ihrer Struktur und Handhabung völlig den auf die herkömmliche Weise erstellten. Sie können auf beliebige Bereiche des Speichers geladen werden, bei freier Wahl des Interpretiersystems. Bei der Entwicklung der ‚macosa'-Quellensprache und des Assemblers waren folgende Schwerpunkte zu berücksichtigen:

– Einsatz für maschinenorientierte Programmierung ohne Einschränkung der von der Gerätetechnik und den Interpretiersystemen gegebenen Möglichkeiten;
– leichte Erlernbarkeit der Quellensprache sowie einfaches Testen der Objektprogramme;
– uneingeschränkte Nutzbarkeit vorhandener Systemunterlagen;
– Eignung für die Bearbeitung wissenschaftlich-technischer und ökonomischer Probleme;
– hoher Bedienkomfort des Assemblers.

1. Quellensprache

1.1. Struktur

Ein in der Quellensprache ‚macosa' geschriebenes Programm besteht aus einer Folge von Informationen mit folgender allgemeiner Struktur:

NAME KENNZEICHEN WORT

1.2. Namen

Ein Name wird aus einer Folge von Buchstaben und Ziffern, beginnend mit einem Buchstaben, gebildet. Es werden nur die ersten fünf Zeichen verarbeitet. Groß- und Kleinbuchstaben haben gleiche Bedeutung. Die Namen r1, r2, r3, r4 sind fest vereinbart und den absoluten Adressen 7773, 7776, 7774, 7745 zugeordnet.

1.3. Kennzeichen und Wort

Allgemein sind folgende ‚Erste Kennzeichen' zugelassen:

0 1 2 3 4 5 6 7 + /

Die Kennzeichen sind in zwei Gruppen einzuteilen. Die Bedeutung der Kennzeichen 3, 4, 5, 6, 7 und + richtet sich nach dem für das Objektprogramm zu verwendenden Interpretiersystem. Vorausgesetzt wird lediglich, daß dem Kennzeichen nur ein zur laufenden Abspeicherung bestimmtes Wort nachgestellt ist.

Kennzeichen und Wort werden unverändert ins Objektprogramm übernommen. Die Kennzeichen 0, 1, 2 und / haben die in allen Interpretiersystemen des C 8205 übliche Bedeutung.

1.3.1. K1 = 0 — Oktalwort

– Die Information kann nach K1 = 0 abgeschlossen werden (leeres Wort).
– Für K2 = 0 (!) 9 werden Kennzeichen und Wort ohne Änderung ihres Informationsgehaltes in das Objektprogramm übernommen.
– Das Kennzeichen K2 = s weist auf den nachgestellten symbolischen Adreßteil. Die letzten drei Zeichen (Ziffern!) des Wortes bilden den Operationsteil des Befehls und werden unverändert ins Objektprogramm übernommen.

Der Adreßteil kann sein

1. eine mit einem Buchstaben beginnende alphanumerische Zeichenfolge. Sie wird als Name eines Operanden oder Befehls interpretiert und durch dessen Maschinenadresse, bei gleichzeitiger Änderung von K2, ersetzt;
2. eine Oktalzahl mit dem Vorzeichen ‚Plus' oder ‚Minus', entsprechend den allgemein üblichen ‚Sternadressen'. Mit Hilfe der ‚Sternadresse' kann auf eine Information relativ zu dem Befehl mit der ‚Sternadresse' Bezug genommen werden. Es gilt:

Adreßteil : = Maschinenadresse des Befehls ± Oktalzahl

1.3.2. K1 = 1 — Sofortmaschinenbefehl

Kennzeichen und Wort werden unverändert ins Objektprogramm übernommen.

1.3.3. K1 = 2 — Hilfsadresse

Für K2 = 0 wird das nachgestellte Wort als Relativadresse, für Kp = 1 als Hauptleitadresse interpretiert. Kennzeichen und Wort werden ohne Änderung ihres Informationsgehaltes ins Objektprogramm übernommen.

1.3.4. K1 = /

Das Kennzeichen wird ins Objektprogramm übernommen. Außerdem wird die Adresse der ersten freien Zelle (Platznummer bzw. Hauptspeicherzelle) ausgedruckt. Die Übersetzung wird fortgesetzt.

1.4. Korrekturen

Die Folge der Korrekturinformationen enthält Korrekturanweisungen und in das Quellenprogramm zu übernehmende Informationen. Die Korrekturanweisungen entsprechen in ihrer Struktur der allgemeinen Struktur der ‚macosa'-Informationen. Folgende Korrekturanweisungen sind möglich

(Na = Name, Oz = Oktalzahl)

– Einfügen

‚Minus' ‚e' Na ‚Plus' Oz

Der Anweisung können beliebig viele Informationen folgen. Sie werden nach der durch Na ‚Plus' Oz gekennzeichneten Information des Quellenprogramms eingefügt.

– Streichen

‚Minus' ‚s' Na 1 ‚Plus' Oz 1
Na n ‚Plus' Oz n

Beginnend mit der durch Na 1 ‚Plus' Oz 1 gekennzeichneten Information, werden bis einschließlich der durch Na n ‚Plus' Oz n gekennzeichneten Information, werden bis einschließlich der durch Na n ‚Plus' Oz n gekennzeichneten Information alle Informationen des Quellenprogramms übergangen.

– Austauschen

‚Minus' ‚a' Na 1 ‚Plus' Oz 1
Na n ‚Plus' Oz n

Die durch Na 1 ‚Plus' Oz 1 und Na n ‚Plus' Oz n begrenzte Informationsfolge wird gegen die der Korrekturanweisung nachgestellte Informationsfolge ausgetauscht. Die Anzahl der einzufügenden Informationen kann ungleich der Anzahl der gestrichenen sein.

– Ende der Korrekturinformationen

‚Minus' ‚Doppelpunkt'. Alle noch nicht verarbeiteten Informationen des Quellenprogramms werden dupliziert. Die Reihenfolge der Korrekturanweisungen muß mit der Reihenfolge der anzusprechenden Informationen des Quellenprogramms übereinstimmen. Im Sinne der Vereinfachung sind Abweichungen von der angegebenen Form der Korrekturanweisungen zulässig. Der Teil eines Quellenprogramms vor dem ersten Namen kann ebenfalls verändert werden.

2. Assembler

2.1. Aufbau und Arbeitsweise

Das relativ adressierte Hauptprogramm arbeitet auf der Basis aller standardmäßigen Interpretiersysteme, da diese nur zum Laden und Aktivieren benötigt werden. Als Rechner sind die Maschinen vom Typ C 8205/6 und C 8205 Z geeignet. Die Übersetzung eines Quellenprogramms erfolgt in zwei Läufen. Dabei arbeitet der Assembler analog zu den die Eingaberegie übernehmenden Teilen der üblichen Interpretiersysteme. Er verlangt jedoch nicht die Angabe einer Haupteinsprungadresse im Quellenprogramm. Ohne Angabe von Haupteinsprungadressen entsteht ein relativ adressiertes Objektprogramm. Als Befehlszähler wird die relative Stellung des Operanden bzw. Befehls zum ersten Wort angegeben. Die Numerierung beginnt mit 0000. Nach Angabe einer Haupteinsprungadresse entsteht ein absolut adressiertes Objektprogramm. Der Befehlszähler gibt den Platz des Befehls bzw. Operanden auf dem Speicher an. Während der Übersetzung wird das Quellenprogramm auf etwaige syntaktische Fehler überprüft. Erkannte Fehler werden gemeldet; die Übersetzung wird fortgesetzt. Dabei wird in das Objektprogramm an Stelle der fehlerhaften Information ein leeres Wort eingefügt. Eine Ausnahme bilden unzulässige Korrekturanweisungen. Sie führen zur Unterbrechung, die Übersetzung kann mit der Eingabe der ursprünglich fehlerhaften Anweisung fortgesetzt werden. Folgende Rechnerläufe sind aufrufbar:

— Erster Übersetzungslauf. Im ersten Lauf ordnet der Assembler jedem Namen des Quellenprogramms eine numerische Adresse zu (Aufbau des Adreßbuches).

— Zweiter Übersetzungslauf. Im zweiten Lauf werden die symbolischen Adreßteile durch numerische ersetzt, verbunden mit einer Änderung des Zweiten Kennzeichens'. Das Objektprogramm wird über Kanal 1 ausgestanzt. Das Adreßbuch bleibt unverändert.

— Korrekturlauf. Das Quellenprogramm wird durch Einfügen, Streichen und Austauschen von Informationen korrigiert und auf Lochband ausgegeben.

— Kombinationslauf. Dieser Lauf ist die Zusammenfassung von Korrektur- und erstem Übersetzungslauf.

— Fortsetzung. Der vorher vereinbarte Übersetzungs- bzw. Korrekturlauf wird fortgesetzt. Dadurch ist es z. B. möglich, ein aus mehreren Teilen bestehendes Quellenprogramm zu einem Objektprogramm zusammenzufassen bzw. nach Eingabefehlern die Übersetzung fortzusetzen.

2.2. Übersetzungsprotokoll

Erkannte syntaktische Fehler werden über Schreibwerk ausgewiesen. Dabei folgt dem fehlerhaften Namen (1. Lauf) bzw. der fehlerhaften Information (2. Lauf) eine mnemotechnische Abkürzung für die Art des Fehlers. Zusätzlich kann für das gesamte Programm oder Teile davon ein vollständiges Übersetzungsprotokoll erzeugt werden. Im ersten Übersetzungslauf bzw. Kombinationslauf werden die Namen und die ihnen zugeordneten numerischen Adressen gedruckt. Im zweiten Übersetzungslauf werden für jede Information der Stand des Befehlszählers, das Objektprogramm und das Quellenprogramm in übersichtlicher Form ausgegeben. Diese Ausgabe kann wahlweise über die Kanäle 0, 1, 2 oder 3 erfolgen, bei Ausgabe über Kanal 1 wird das Objektprogramm unterdrückt. Die Lochbandausgabe setzt die Übersetzungszeiten erheblich herab und bietet die Möglichkeit, unabhängig vom Rechner beliebig viele Protokollexemplare herzustellen.

2.3. Problemgröße

Der Anzahl der Informationen im Quellenprogramm wird vom Assembler keine Grenze gesetzt, die Anzahl der Namen ist auf 1300 (dezimal) beschränkt.

3. Objektprogramm

Das im Ergebnis der Übersetzung vorliegende Objektprogramm kann zur Abarbeitung auf beliebige Speicherbereiche geladen werden, wenn es nicht durch Informationen mit K1K2 = 21 oder seine Struktur (zeitoptimal, Eingabe-Halt-Befehle) an bestimmte Bereiche gebunden wurde. Bezüglich des Befehlsvorrates und der Möglichkeiten oder zeitoptimalen Programmierung gibt es keine Einschränkungen. Interpretiersystem und Speicherbelegung werden vom Programmierer gewählt. Der Assembler wird zur Abarbeitung der Objektprogramme nicht benötigt. Der MACOSA-Einsatzbereich umfaßt die Programmierung auf der Grundlage maschinencode-orientierter Interpretiersysteme (MEP, SACK, BIS, Gis) sowie die ‚Programmierung im Maschinencode' für echte Interpretierer (GIPS, FIPS) und Compiler (PS2, FORMAL).

4. Beispiel

Es soll die Summe S = C [1] + C [2] + ... + C [n] ermittelt werden! Die Zahlen C [i] sind als Dezimalzahlen satzweise (Marke ‚Doppelpunkt'; ‚WRZL' ohne Bedeutung) zur Verfügung zu stellen und bei der Eingabe im Anschluß an die Befehls-

macosa2

```
0000  004600030              00   4600 030
0001  0130211                0s   sb  211
0002  007745240              0s   r4  240
0003  0131240                0s   aci 240
0004  007566025              00   7566 025
0005  014000                 0s   -1  000
0006  007745211              0s   r4  211
0007  0133220                0s   a1  220
0010  0132240                0s   vk  240
0011  0134211                0s   c1  211
0012  007773240              0s   r1  240
0013  0117000                0s   +4  000
0014  00251          zykl    00        251
0015  007776240              0s   r2  240
0016  007553025              00   7553 025
0017  0131211                0s   aci 211
0020  0133210                0s   a1  210
0021  0131240                0s   aci 240
0022  0132330                0s   vk  330
0023  0134140                0s   zykl 140
0024  004500030              00   4500 030
0025  002720030              00   2720 030
0026  002077561025           00   2077561 025
0027  00020                  00        020
0030  0134240          sb    0s   c1  240
0031  0                aci   0
0032  0                vk    0
0033  001000           a1    00        1 000
0034  /                c1    /
```

Abb. 1: Übersetzungsprotokoll

folge zwischenzuspeichern! Abbildung 1 zeigt das vollständige Übersetzungsprotokoll des zweiten Laufes. Die Zeichenfolge in der 2. Spalte (Objektprogramm) stimmt mit der auf dem Lochband überein und wird dort mit der Wortmarke ‚Punkt' abgeschlossen (Ausnahme: K1 = /). Das Quellenprogramm wurde vom Assembler aufbereitet.

Das Assemblerprogramm wird seit geraumer Zeit erfolgreich eingesetzt. Es trägt wesentlich zur schnellen und bequemen Erstellung von Programmen bei. Eine Weiterentwicklung des Assemblers, etwa mit der Zielstellung, auch bei Befehlen mit symbolischem Adreßteil über die vorderen 4 Triaden des Maschinenwortes zu verfügen, ist bei Bedarf möglich. Detaillierte Informationen stellt das ORZ der Martin-Luther-Universität, 402 Halle, Weinbergweg, gern zur Verfügung.